REMARKS

This paper is submitted in reply to the Office Action dated March 20, 2008, within the three-month period for response. Reconsideration and allowance of all pending claims are respectfully requested.

In the subject Office Action, claim 15 was rejected under 35 U.S.C. §101 as non-statutory subject matter. In addition, claims 1, 3-8, 10-16 and 18-21 were rejected under 35 U.S.C. §103(a) as being unpatentable over the article "Efficient Mid-Query Re-Optimization of Sub-Optimal Query Execution Plans" by Kabra et al. (hereinafter Kabra) in view of U.S. Patent Publication No. 2005/0177557 to Ziauddin et al (hereinafter Ziauddin).

Applicants respectfully traverse the Examiner's rejections to the extent that they are maintained. Applicants have canceled claims 13-16 and 18-21 herein. Applicants note that the cancellations made herein are being made only for facilitating expeditious prosecution of the aforementioned claimed subject matter. Applicants are not conceding in this application that the originally claimed subject matter is not patentable over the art cited by the Examiner, and Applicants respectfully reserve the right to pursue this and other subject matter in one or more continuation and/or divisional patent applications.

Turning to the rejections of claims 13-16 and 18-21, the §103(a) rejections of these claims and §101 rejection of claim 15 have been rendered moot by the cancellation, and should be withdrawn.

INDEPENDENT CLAIM 1

Now turning to the subject Office Action, and specifically to the Examiner's §103(a) rejection of independent claim 1, this claim generally recites a method for automatic handling of errors within a database engine. The method includes detecting an error while executing a query access plan. The error is an execution error of a type that halts execution of the query access plan, and the query access plan is of the type generated by a query optimizer. The method also includes, in response to detecting the error, automatically rebuilding the query access plan with the query optimizer to generate a new query access plan, and executing the new query access plan to generate at least a portion of a result set for storage or display.

Claim 1 relates to an **execution error of a type that halts execution of the query access plan**. As described in page 10, lines 1-6 of the Application, during execution, the database engine 44 may, in step 304, detect an error that halts execution of the query. Within some database environments, such an error is known as a function check. A number of different such errors may be detected; some relate to opening various tables; some relate to formatting the result set; some relate to problems with selection criteria; and many other types of errors are possible as well. Moreover, as indicated on page 12, lines 21-23 of the Application, execution errors sometimes occur because of problems in the database engine 44 or the optimizer 42, with a "patch" typically issued from the database vendor that addresses the problems. Nonetheless, the claimed execution errors halt execution of a query, and are generally involuntary operations that essentially halt a query and halt its continued execution.

In rejecting claim 1, the Examiner once again relies on Kabra, and now combines Kabra with a new reference, Ziauddin. The Examiner asserts that Kabra discloses limitations of the claim 1 at the abstract, lines 6-8; page 109, column 2, lines 34-37; page 110, column 1, lines 10-15; page 109, column 2, line 34-page 110, column 1, line 4; page 110, column 1, lines 2-4 and lines 13-15; and page 110, column 1, line 15. The Examiner admits, however, that Kabra fails to explicitly disclose the limitation **"wherein the error is an execution error of a type that halts execution of the query access plan"** (emphasis added).

For this, the Examiner cites Ziauddin in the abstract and in paragraphs [0015] and [0017]. The Examiner also asserts that Ziauddin also discloses "in response to detecting the error, automatically rebuilding the query access plan to generate a new query plan" in paragraph. [0017].

However, contrary to the Examiner's assertion, Ziauddin adds nothing to the rejection because, just like Kabra, Ziauddin does not teach or suggest **an execution error that halts execution of a query access plan**. The errors that are addressed in Ziauddin are runaway errors, which result when a query runs longer than expected (paragraph [0017]).

Indeed, if anything the runaway errors of Ziauddin are the precise opposite of an "execution error that halts execution of a query access plan." As indicated in paragraph [0004] of Ziauddin, a "...supoptimal plan results in a run-away execution of the

query…[in] other words, the plan, when executed, causes a SQL statement to run for a long time with enormous use of system resources." Therefore, runaway errors continue execution, rather than halt execution, as is required by claim 1. There is nothing associated with a runaway error that causes the execution of a query access plan to be halted. In fact, a runaway error prevents a query access plan from stopping or completing.

In addition, the disclosure of Ziauddin is more inline with optimizing sub-optimal query plans, like Kabra, because Ziauddin monitors these runaway errors, and only the longest running queries may be selected for reoptimization, leaving the others to continue to execute. Indeed, paragraphs [0003]-[0004] disclose the problem of suboptimal plans, and the Abstract and paragraph [0017] go on to disclose an automatic tuning optimizer (ATO) that, in a background process, optimizes the execution plan for queries having runaway execution. First, the query executions having a negative execution time difference can be automatically identified as a runaway query execution. Next, the ATO performs various analyses of the corresponding SQL statement such as automatic identification and correction of incorrect statistics, cardinality estimates, and cost estimates related to the statement. The ATO builds a new plan, compares the estimated execution time of the new plan to the remaining execution time of the current plan. According to paragraph [0018], the remaining execution time estimate for the current plan and the execution time estimate for the new plan should be more accurate because these figures are now based on validated estimates due to the execution of the query thus far. If the estimated time for the new plan is less than the remaining time for the current plan, the current plan is aborted and replaced with the new plan.

Thus, while the Examiner does correctly point out that the Abstract of Ziauddin teaches execution of query plans and paragraph [0017] teaches aborting a current plan and replacing it with a new plan, both of these features are in the context of runaway errors that continue the execution of a query, and NOT execution errors that halt execution of a query and the associated access query plan, as required by claim 1.

It may be that the Examiner is taking the position that since Ziauddin discloses aborting a query due to a runaway error, this somehow discloses an execution error that halts the execution of a query. However, claim 1 requires that the "error" halt the execution of a query. In Ziauddin, the optimizer aborts a query after selecting an

appropriate candidate, so it is the optimizer that actually halts the query.

Indeed, the aborting and substitution for the new plan in Ziauddin is more or less voluntary, based on statistics and estimated times, because execution of a runaway query can and will often continue with the current plan as indicated in block 160 and paragraphs [0015]-[0017], albeit the execution may take longer. Block 160 of FIG. 1 indicates, that the process "determine[s] whether to abort the current plan and execute the new plan". Moreover, paragraphs [0015] and [0017] indicate that the current plan MAY be aborted but does not need to be. For instance, paragraph [0015] states that "[t]he automatic prevention of run-away query executions may abort a current execution of a query run if the automatic process has produced an improved plan in the background, and further, has determined a benefit to aborting the current execution and performing an execution of the new plan." Thus, aborting the current is not automatic.

It also is important to note that the early exit by aborting stems from runaway errors causing the extended execution of runaway queries due to nonvalidated statistics, and NOT from the actual execution of a query. Furthermore, any error in statistics collection leading to the runaway errors does not cause a query access plan to be automatically rebuilt, as is required by claim 1.

Moreover, even if an access plan is aborted in Ziauddin, it is because the runaway error causing the query to continue to execute for too long and using too many resources (paragraph [0004]), not because of an execution error that causes the query (and query access plan) to stop executing. Similarly, the detection of a sub-optimal query, as occurs in Kabra, perhaps results in a query being modified or re-optimized, but the decision to perform such a modification or optimization is made by the optimizer, and is more or less a voluntary operation performed on the basis of collected performance statistics. Execution errors that halt execution of a query, on the other hand, are involuntary operations that essentially terminate a query and prohibit its continued execution. And, Applicants can find no teaching in either reference purporting to disclose or suggest specifically performing an automated rebuild of a query access plan "in response to" a detected error that halts execution of the query access plan.

The runaway errors, in fact, do not even appear to be execution-related errors, since even when such "errors" exist, the queries apparently complete at some point in the future.

Were the runaway errors in Ziauddin of the type that caused a query execution engine to "hang" and become unresponsive, these errors might arguably construed as "execution" errors. However, given the runaway errors are only associated with sub-optimal query execution, they are not properly considered to be execution errors of the type contemplated by Applicants' claims.

While Kabra and Ziauddin both generally attempt to improve optimization of database queries, neither reference, alone or in combination, discloses or suggests doing so for "an execution error of a type that halts execution of the query access plan", among other limitations in claim 1. Applicants accordingly submit that the combination proposed by the Examiner does not disclose or suggest each and every limitation in claim 1, and therefore, Applicants respectfully submit that independent claim 1 is novel and non-obvious over Kabra and Ziauddin. Reconsideration and allowance of claim 1, and of claims 3-6 which depend therefrom, are therefore respectfully requested.

INDEPENDENT CLAIM 7

Turning next to the Examiner's §103(a) rejection of independent claim 7, claim 7 generally recites a method for automatic handling of errors within a database engine. This method includes receiving an error while executing a function within a query access plan. The error is an execution error of a type that halts execution of the query access plan, and the query access plan is of the type generated by a query optimizer. The method also includes identifying a first implementation method of the function within the query access plan, rebuilding the query access plan with the query optimizer by replacing the first implementation method with a second implementation method of the function so as to generate a new query access plan, and executing the new query access plan to generate at least a portion of a result set for storage or display.

In rejecting claim 7, the Examiner also relies on the same arguments and citations of Kabra and Ziauddin made in connection with claim 1. The Examiner admits once again, however, that Kabra fails to explicitly disclose the limitation "wherein the error is an execution error of a type that halts execution of the query access plan". The Examiner also admits that Kabra does not disclose "identifying a first implantation method of the function within the new query access plan" and "rebuilding the new query access plan by replacing

the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan". For the latter two limitations, the Examiner cites Ziauddin, and in particular, paragraphs [0029] and [0017].

As argued above, the combination of Kabra and Ziauddin does not disclose an execution error of the type recited in Applicants' claims, and additionally, does not disclose functions and replacing the first implementation method with a second implementation method of the function.

First, paragraphs [0029] and [0017] of Ziauddin merely disclose statistics and profiling for generating the new access plan in the background by the ATO. However, these paragraphs do NOT teach or suggest any functions or the function related implementations. Statistical and profiling information are helpful and often used to select an implementation method, and may even be stored in an implementation method (e.g., in an index), but statistical and profiling information are NOT implementation methods. As described in page 12, lines 8-24 of the Application:

> In a robust database system there are hundreds of different query functions with alternative implementation methods that an optimizer chooses from when building a query plan. Even though an exhaustive list of the different functions and their alternative implementation methods are not explicitly included herein, the above-described method contemplates, when appropriate, rebuilding a query plan with alternative implementation methods for any of the wide variety of functions likely to be encountered in a query plan. For example, an ordering function may be implemented using "order by index" or "order by sort"; a grouping function may be implemented using "group by hash" or "group by index"; and a join function may be implemented using either a nested loop method or a hashing method. Additionally, when handling a subquery function, the optimizer may implement it as a "join" or as a separate subquery; or when coding a view function in the query plan, the optimizer can implement the view function using a method known as "materialization" or an alternative method known as "merge". Other exemplary functions include such functions as hash, view, host variables, parameter markers, literals, nested

loop, materialization, sort, bit map, sub query, sparse index, merge, and table scan. These are just a few exemplary functions for which an optimizer selects from among alternative implementation methods when generating a query plan.

Second, there is no disclosure in Ziauddin of replacing implementation methods related to functions. For example, cited paragraph [0017] explicitly indicates that "if the execution plan built by the ATO is different from the one that is currently executing, the ATO can estimate…", which implies that the new access plan created by the ATO is NOT always different from the current plan. And if the new access plan generated by the ATO is not different, then the same implementation methods are being used, therefore, there is no replacement of a first implementation method with a second implementation method. In instances that the new access plan is different from the current access plans, the implementation methods can be different but there is no disclosure in Ziauddin affirmatively teaching or suggesting this, let alone with respects to functions.

Indeed, while Kabra and Ziauddin both generally attempt to improve optimization of database queries, neither reference, alone or in combination, discloses or suggests "wherein the error is an execution error of a type that halts execution of the query access plan"; "identifying a first implantation method of the function within the new query access plan"; and "rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan". Applicants accordingly submit that the combination proposed by the Examiner does not disclose or suggest each and every limitation in claim 7, and therefore, Applicants respectfully submit that independent claim 7 is novel and non-obvious over Kabra and Ziauddin. Reconsideration and allowance of claim 7, and of claims 8 and 10-11 which depend therefrom, are therefore respectfully requested.

INDEPENDENT CLAIM 12

Turning next to the Examiner's §103(a) rejection of independent claim 12, claim 12 generally recites a method for automatic handling of errors within a database engine. The method includes executing a query access plan comprising a plurality of functions, each function including a first implementation method, and the query access plan of the type

generated by a query optimizer, detecting a first error when executing a first function. The first error is an execution error of a type that halts execution of the query access plan. The method also includes rebuilding the query access plan with the query optimizer to generate a new query access plan, executing the new query access plan to generate at least a portion of a result set for storage or display, receiving a second error while executing the first function within the new query access plan, and rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function.

In rejecting claim 12, the Examiner also relies on the same arguments and citations of Kabra and Ziauddin made in connection with claims 1 and 7. The Examiner once again admits, however, that Kabra fails to explicitly disclose the limitation "wherein the error is an execution error of a type that halts execution of the query access plan". The Examiner also admits that Kabra does not disclose "receiving a second error while executing the first function within the new query access plan" and "rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function". For these latter two limitations, the Examiner cites Ziauddin, and in particular, paragraphs [0029] and [0017].

In addition to Applicants' arguments from claim 1, as argued in connection with claim 7, paragraphs [0029] and [0017] do not teach or suggest any functions or function related implementations, let alone teaching or suggesting a plurality of functions, a second function, and additional function related implementations for the second function. This claim generally recites in part the detection of a second error occurring during the execution of a rebuilt query access plan, and the handling of such an error by rebuilding the query access plan to replace a first implementation method of a function with a second implementation method of the function. Applicants can find no disclosure or suggestion in either Ziauddin or Kabra directed to any type of multi-stage error processing method that handles subsequent errors that occur after an attempt has already been made to rectify an error in a query access plan, much less any method that handles a subsequent error in the specific manner recited in claim 12.

Even assuming *arguendo* that the handling of errors in Ziauddin and Kabra is analogous to automatically rebuilding a query access plan in response to a detected error,

as asserted by the Examiner, neither reference addresses any functionality for handling a subsequent error that occurs after a query access plan has been rebuilt and re-executed. Accordingly, Applicants submit that claim 12 is additionally patentable over the cited references for this additional reason.

Indeed, while Kabra and Ziauddin both generally attempt to improve optimization of database queries, neither reference, alone or in combination, discloses or suggests "wherein the error is an execution error of a type that halts execution of the query access plan"; "identifying a first implantation method of the function within the new query access plan"; "rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan"; "receiving a second error while executing the first function within the new query access plan"; and "rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function" Indeed, independent claim 12 has additional function related limitations that the Examiner cannot ignore.

Applicants accordingly submit that the combination proposed by the Examiner does not disclose or suggest each and every limitation in claim 12, and therefore, Applicants respectfully submit that independent claim 12 is novel and non-obvious over Kabra and Ziauddin. Reconsideration and allowance of claim 12 are therefore respectfully requested.

DEPENDENT CLAIMS

Applicants also traverse the Examiner's rejections of the dependent claims based upon their dependency on the aforementioned independent claims. Nonetheless, Applicants do note that a number of these claims recite additional features that further distinguish these claims from the references cited by the Examiner.

For example, claim 3 depends from claim 1 and additionally recites that the error is a function check. Examiner cites Kabra at page 109, column 2, lines 29-33. However, the cited disclosure at page 109 does not disclose a function check, which is recognized in the art as a type of error that halts execution in a database engine. Moreover, as argued above in connection with claims 7 and 12, Ziauddin also does not teach or suggest functions or anything related to functions. Furthermore, even if the Examiner attempts to argue that

the runaway errors of Ziauddin somehow halt execution of a query because an optimizer may voluntarily select a runaway query to be aborted, these errors are not properly classified as function check errors (and indeed, are completely opposite thereto because they continue to run longer than expected). Reconsideration and allowance of claim 3 are therefore respectfully requested for this additional reason.

Claims 4 and 10, which depend from claims 1 and 7, respectively, recite to varying extents the concept of handling another error detected while executing a query access plan that has been automatically rebuilt in response to an execution error by rebuilding the query access plan to replace a first implementation method of a function with a second implementation method. Claim 4 is representative, and recites receiving another error while executing a function within the new query access plan, identifying a first implementation method of the function within the new query access plan, and rebuilding the new query access plan by replacing the first implementation method with a second implementation method of the function so as to generate a rebuilt query access plan.

In rejecting claim 4, the Examiner relies on paragraph [0029], relating to the creation of the new access plan by the ATO. It is created from the statistics, estimates, and settings analyses to be stored in a SQL Profile. Once the SQL Profile is created, it is used in conjunction with the existing statistics by the compiler to produce a well-tuned plan for the corresponding SQL statement. As argued above in connection with claim 7, paragraph [0017] implies that the new access plan may be the SAME as the current plan, with the same implementation methods, therefore, there is teaching or suggestion of replacing the implementation method. Moreover, Ziauddin does not disclose the function related limitations. Reconsideration and allowance of claims 4 and 10 are therefore respectfully requested for this additional reason.

Claim 8 depends from claim 7, and additionally recites that the function is one of a join function, an indexing function, a grouping function, and an ordering function. In rejecting claim 8, the Examiner relies on page 109, column 2, lines 29-33 of Kabra. As argued in connection with claim 3, column 2 does not teach or suggest function checks.

Although column 2 mentions joins, it is in the context of improving a sub-optimal query plan. Indeed, Section 2.4 of column 2 explicitly says, "the join order might be sub-optimal, or the choice of algorithms (e.g., hash-join vs. indexed nested-loops join) could be

improved." There is NO error with the joins in Kabra that warrants replacing the implementation method. If replaced, it is to merely improve a sub-optimal query plan, not because of the claimed limitations.

Moreover, the rest of column 2 deals with relatively simple changes to improve the execution of a query, such as modifying the allocation of memory to the various operators in the query without actually modifying the query execution plan. Ziauddin also does not disclose the function related limitations. Reconsideration and allowance of claim 8 are therefore respectfully requested for this additional reason.

Claims 5 and 6, and claim 11, which depend from claims 1 and 7, respectively, recite to some extent logging information about at least one error and the new query access plan. Claim 11 is representative, and recites logging information about the error, the another error, and the new query access plan.

In rejecting claim 11, Examiner relies on page 109, column 1, lines 16-27 of Kabra, relating to gathering statistics about intermediate query results during the course of query execution to be used to improve the estimates of the query optimizer. These improved estimates can be used to improve the allocation of memory to the various operators of the query.

Kabra indicates that when improved estimates are available, the memory management module can be re-invoked and supplied with the new estimates. The memory management module used these new estimates to produce a new memory allocation for the remainder of the query. Overall performance is expected to improve since the new memory allocation is based on improved estimates.

However, none of this discloses logging information about errors and the new query access plan. Logging estimates or statistics is NOT the same as logging information about errors and the query access plan. Reconsideration and allowance of claims 5-6 and 11 are therefore respectfully requested for this additional reason.

In summary, Applicants respectfully submit that all pending claims are novel and non-obvious over the prior art of record. Reconsideration and allowance of all pending claims are therefore respectfully requested. If the Examiner has any questions regarding the foregoing, or which might otherwise further this case onto allowance, the Examiner may contact the undersigned at (513) 241-2324. Moreover, if any other charges or credits

are necessary to complete this communication, please apply them to Deposit Account 23 3000.

Respectfully submitted,

June 20, 2008
Date

/ Scott A. Stinebruner /
Scott A. Stinebruner
Reg. No. 38,323
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234